A Software Defined Device Provisioning Framework Facilitating Scalability in Internet of Things

Alex Mavromatis[1], Aloizio Pereira Da Silva[1], Koteswararao Kondepu[1],
Dimitrios Gkounis[1], Reza Nejabati[1], Dimitra Simeonidou[1]
[1]High Performance Networks Group, University of Bristol, Bristol, United Kingdom,
Emails: {a.mavromatis, dimitra.simeonidou}@bristol.ac.uk

***Abstract*—The evolution of communication technologies is growing faster than ever demanding efficient networking operations. 5G will soon be the new generation of telecommunications and it is able to unlock the future design challenges of Internet of Things (IoT). Low latency, deployment scalability, automatic maintenance and high bandwidth are some of the core elements accommodating future efficient IoT deployments.**

In this paper a novel architecture and implementation based on Software Defined Networking (SDN) principles is presented aiming to solve scalability issues within large IoT deployments. This innovative solution is capable of reducing the latency, improving reliability and the User Experience (UX) within the IoT infrastructure. The proposed framework, named Software Defined Provisioning (SDP) enables scalability within IoT deployments by enhancing the network with plug and play features. Furthermore, the integration of an IoT platform with SDN is enabling the robust dynamic authorization and provisioning of heterogeneous IoT devices. The performance evaluation shows that our proposed framework significantly improves the current provisioning systems and also empowers diverse IoT platforms with a plug-able SDN interface.

I. INTRODUCTION

One of the key enablers of Internet of Things (IoT) is integration. Therefore, equipping IoT with other technologies is a feasible way of enabling large deployments and also making the process secure [1]. IoT deployments are happening for both industrial and research purposes, nevertheless there are several obstacles that need to be addressed for successful large deployments. One fundamental yet very critical barrier in IoT is the efficient management and provisioning of the devices. Successful scalability of IoT deployments require dynamic provisioning of the devices (mostly automated) with minimal administrative effort.

Therefore, the fundamental requirement of fifth generation (5G) technology [2] is to support the aforementioned issues by providing a smarter network. However, 5G could possibly solve scalability, reliability and latency challenges, it is not still clear how these things can be addressed, as 5G is under standardization [3]. As stated in [4] device provisioning solutions are expected to have a large impact from 2016 to 2021, as more and more companies are adopting provisioning solutions to manage and control the identities and accesses of IoT applications and network resources.

According to [5], scalability in IoT is critical, especially in large deployments where low power devices are not easily monitored. Thus, this paper presents a design and implementation of a promising framework for massive IoT deployments based on Software Defined Networking (SDN) principles, namely Software Defined Provisioning (SDP). The proposed SDP framework is integrating a standard SDN controller with an open source IoT platform.

The SDP framework introduces a novel approach to provision IoT devices within a data center infrastructure for authorization and connectivity reasons. Moreover, SDP assists an IoT network with scalability, especially within large Wireless Sensor Network (WSN) deployments where operation costs are critical. Furthermore, improving core IoT elements such as latency, information reliability and User Experience (UX) can facilitate new use cases in IoT 5G networking.

More specifically, the SDP framework provides a provisioning system for IoT platforms, where each device sending data to services needs to be provisioned and authorized in order to send data. Here, the framework operates as a continuous service filtering the SDN topology information (i.e., Media Access Control (MAC) address and Internet Protocol (IP) address) which is stored and used as a benchmark for the device provisioning.

Currently the provisioning happens through manual configuration or software provisioning by utilizing predefined knowledge of the devices identification. To the best of our knowledge, SDP is the first IoT device provisioning framework that has the ability to automatically enable IoT devices and provide the provisioning through SDN. The results presented in Section VI show that SDP is a promising architecture for solving scalability issues in the large IoT deployments.

The remainder of this paper is organized as follows. The literature review and related work is presented in Section II. Section III introduces the building blocks of the SDP architecture. Section IV describes the SDP provisioning process. Section V describes the experimental methodology we use to evaluate SDP while Section VI presents our performance evaluation. We conclude the paper with directions for future work in Section VII.

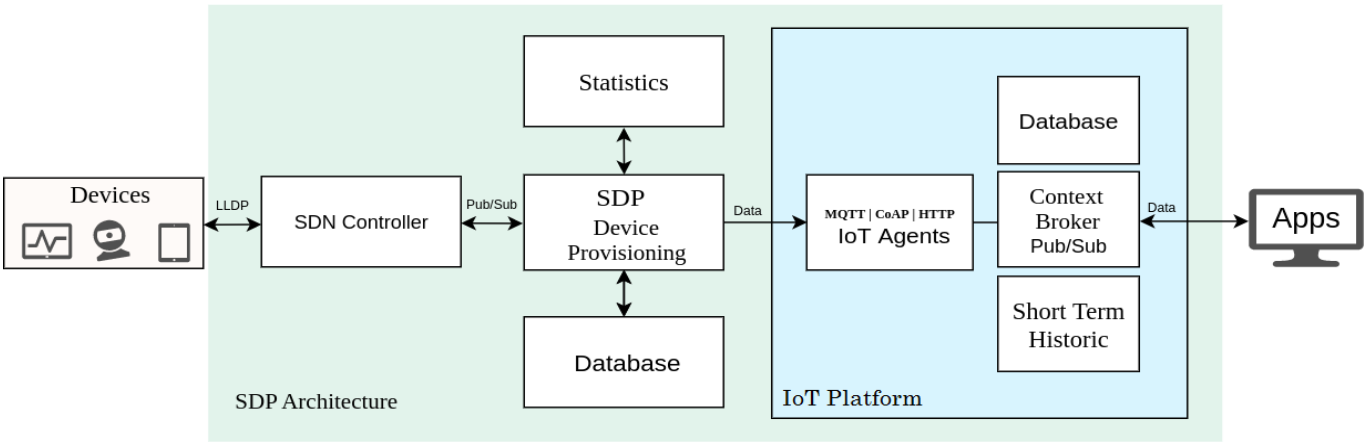


Fig. 1: IoT Provisioning Architecture

II. Related Work

The Internet and its services coupled with ubiquitous wireless communication and increasingly more powerful and smaller mobile devices have enabled anywhere, anytime connectivity and computing. Consequently, applications such as Intelligent Transportation Services, Smart Homes, Smart Cities, are considered part of The Internet of Everything are no longer just a vision in the future, but rather a reality [5].

These "new generation" distributed computing applications will revolve around increased device-to-device communication and cooperation as well as increased flexibility and scalability requirements [6][7][8]. The autonomous attach and detach of these device to heterogeneous networks are fundamentally important from the network's performance point of view enabling users to provision millions of devices in a secure, autonomous and scalable manner. The process of attaching an IoT device to the network is known as "device provisioning". In the context of provisioning IoT devices to the data center infrastructure, provisioning can be divided into two mainly phases, as follows.

1) The first phase is establishing the initial connection between the device and the IoT solution by registering the device.
2) The second phase is applying the proper configuration to the device based on the specific requirements of the solution it was registered to.

The authors proposed in [9] the Open Connectivity Foundation (OCF) [10] a provisioning concept combined with an international standard (ISO/IEC 18092) called Near Field Communication (NFC). NFC enables wireless communication between two devices at close proximity, to enable end-to-end provisioning through cloud services. In SDP we combine the provisioning concept with the IoT platform and network softwarization to provide more flexibility and scalability to the provisioning process.

The idea of network softwarization applied in the IoT context has been presented in [11]. A novel multi-network controller architecture that enables the controller being instantiated multiple times in different location contributing to the scalability. The proposed framework is leveraging an SDN controller to forward device information to the service and automatically retrieve any change about IoT devices.

A software define based framework model, named Software Defined Internet of Things (SDIoT), is described in [12]. In this framework the authors proposed to simplify the IoT management process integrating software defined network, software defined storage and software defined security into the proposed model. The SDIoT's goal is to facilitate IoT control and management operation using software defined networking. The SDIoT framework does not consider IoT device provisioning, however it is focused on IoT data distribution and storage.

A simplified and customizable approach to integrate highly scalable, available, and customizable IoT certificate provisioning and deployment is proposed in [13]. The proposed approach aims to deploy a certificate management platform for IoT devices. However it does not focus on IoT device provisioning process performance improvement but in the authentication process.

III. Architecture

The proposed architecture is a different approach of organizing the provisioning and authorization of devices within an IoT infrastructure. Mainly due to the reason that we combine the IoT platform with the SDN capabilities. By introducing SDN topology discovery concepts and enhancing the IoT platform with SDN control, the current architecture can become more scalable, hence the performance of large deployments can be much faster and efficient.

Making the aforementioned process dynamic utilizing software is assisting scalability in IoT. However we intend to provision at a network layer and provide a dynamic provisioning architecture where the network is taking decisions. Within this architecture the SDP framework operates as a middleware between the devices and the IoT platform leveraging the advantages of SDN.

IoT devices generate data and transmit to the IoT platform endpoint. The networking is being established through the

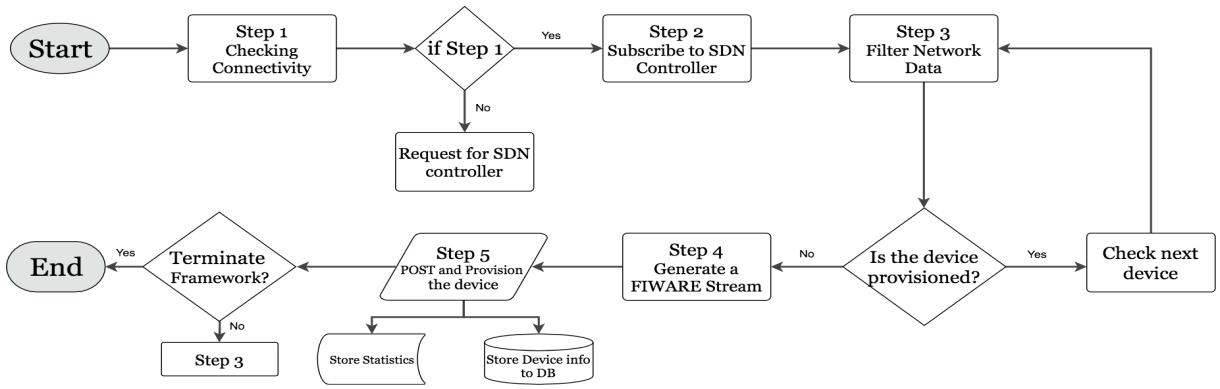


Fig. 2: Software Defined Provisioning Process

SDN controller. Afterwards the SDP framework will filter the device information and will provision them. The applications will use a RESTful Application Programming Interface (API) to collect the information and utilize it based on the use case. Figure 1 shows the IoT provisioning architecture of this system and its main components. The architecture is composed by five main building blocks.

1) **IoT Devices**: Includes any kind of computing device that has one or more smart sensors attached and it is able to connect wirelessly to a network and has the ability to transmit data. For example devices such as air quality sensors, water quality sensors, wearables and actuators.
2) **SDN Controller** : It is a modular open building block that provides network programmability. This building block is directly communicating with the IoT devices via Link Layer Discovery Protocol (LLDP) [14][15].
3) **IoT Platform**: The IoT Platform building block provides a set of APIs that enables the development of Smart Applications in multiple vertical sectors. The specifications of these APIs are public and royalty-free. The IoT provisioning architecture uses an IoT Agent component which enables the communication between other IoT platform components (i.e. context broker) and the IoT devices. Basically, the IoT agent provides a southbound interface enabling different IoT data protocols (i.e. Message Queue Telemetry Transport–MQTT, Constrained Application Protocol–CoAP and Hypertext Transfer Protocol–HTTP) to communicate with a set of heterogeneous IoT devices. The IoT agent uses two headers to manage IoT device's request: 1) *Service* that represents a tenant to manipulate a specific service; 2) *ServicePath* that enables access to specific information entities retrieve a service. Furthermore the information is also stored to a database and to a component named Short Term Historic that enables short term queries.
4) **Software Defined Provisioning (SDP)**: The proposed SDP framework is a middle-ware component that enables the communication of the IoT platform and the SDN controller. More specifically, SDP creates a stream which contains the IoT device identification, device MAC address, service and servicePath information. This stream will provision the device and establish a secure communication between the device and the IoT platform. Also the framework stores statistics and all the functionality of the provisioning process to a database.
5) **Applications (Apps)**: This block includes any IoT application that requires device flexibility and scalability. For experimentation purpose, the IoT application is implemented using an emulated WSN that is monitoring environmental values such as temperature, humidity etc. The applications use the publish subscribe pattern to retrieve real time data.

IV. Device Provisioning Framework Implementation

This section illustrates the design, development and implementation of the SDP framework. The functionality of the framework is considered to be a middle-ware between the OpenDayLight (ODL) [16] controller and the FIWARE [17] IoT platform. As SDN controller we selected ODL based on the software maturity and support. This framework requires software development, therefore we select FIWARE for the IoT platform since it is an open-source solution. Thus we are able to extend and integrate these open-source entities based on the generic architecture presented in Section III. The implementation setup requires connectivity between the elements of FIWARE and ODL. Mainly it is a back-end service written in *Node.js* running and monitoring the deployment. The next subsections describe in detail the steps of the provisioning process (Figure 2).

A. Testing connectivity and subscribing to SDN controller

As shown in Figure 2, the process starts (step 1) by checking, the frameworks requirements, the connectivity among SDP the controller and the FIWARE service. Once the connectivity is set then the framework will subscribe to ODL (step 2) for device information and will retrieve any update automatically, seeking information about each device connected to the network. The data received from the back-end service is then passed to a filtering function that analyses and stores the information.

B. Filtering SDN data

As long as the data from the controller is valid and securely passed to the back-end service the next step is data processing (step 3). The identity information of each device is stored into a database. The network addresses are being analyzed and if the new device MAC address matches the previous provisioned devices then the software will continue scanning the information, until a new device is detected. If the device is "new IoT hardware" then the identity information will be passed to a FIWARE constructor function that will provision the device.

C. Hardware Provisioning

The hardware provisioning is a stream message (step 4) sent to the FIWARE service that runs in the data center. This functionality occurs once the framework will be determined that a new device is trying to push data at the FIWARE endpoint. The back-end service has implemented a constructor function which always receives identity device information therefore initializes and sends a stream equivalent of the FIWARE device provisioning system. If FIWARE responds with a success message then the framework will store the information (step 5) of the device in a database for future device validation and statistical analysis.

D. Enabling the framework for other services

The automatic provisioning of the devices is a continuous service ensuring that each device inserted will be provisioned and able to send information instantly. The integration of SDN and IoT platforms generate a lot of features that could improve the WSN functionality. The SDP information interface enables alternate to FIWARE also other IoT platforms in order to tackle issues of software attachment to the core by providing a plug and play philosophy.

In addition, the framework accommodates other end-user applications by making the information available throughout a RESTful API interface. Additional network tools or services are able to collect the information generated and utilize it for the decision making, therefore the integration and improvement of IoT services are capable to expand leveraging SDP.

V. Experimentation environment

A. Experimentation Setup

This section presents a proof of concept based on the architecture described in Section III. The experimentation environment is comprised of two different Virtual Machines (VMs) where ODL controller and FIWARE platform are running. An Ubuntu VM for ODL and a CentOS 6.6 VM are deployed within the network infrastructure where connectivity between them is established. Note that two different operating systems (OSs) are considered because FIWARE platform operates only with CentOS.

The proposed SDP framework is running on the same Ubuntu VM where ODL controller is running and enables the device provisioning. Finally, all the information about the latest provisioned device is being stored into a Database. The data transmission is emulated by implementing the MQTT protocol. Unless specified, the considered software and the experimentation setup details are presented in Table 1.

TABLE I: Experimentation Setup and Software

Topology	**Value**
Topology	Random
Number of devices	100, 200, 300, 400, 500
Experiments	**Value**
Duration	Varies
Traffic Pattern	5000 packets / experiment
IoT protocol	MQTT
Software used	**Version - component**
OpenDayLight	Boron
FIWARE	IoT Agent, Orion Context broker, Short Term Historic
Node.JS	Version 4.2.6

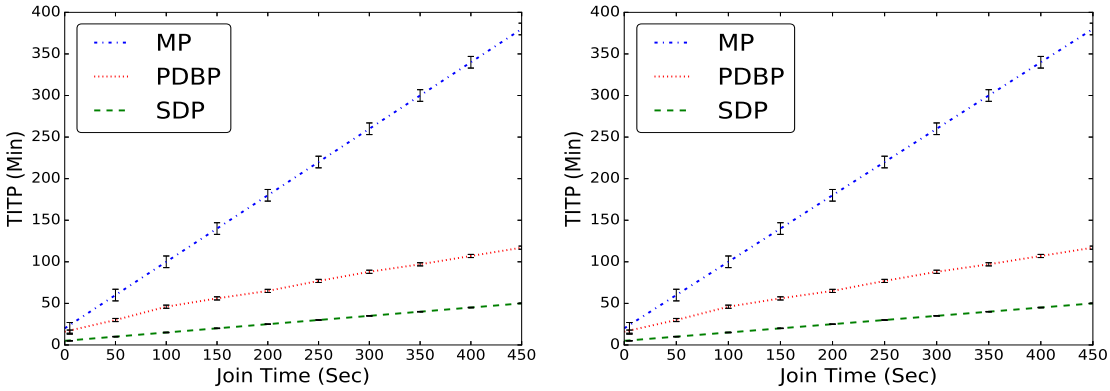
SDP is compared with the currently available two provisioning methods as described below:

- *Manual Provisioning (MP)*: It is a manual process by sending a request to the IoT agent component, the user needs to send a POST request using a terminal within the FIWARE VM. Note that this approach does not provide any scalability and its only feasible for small IoT deployments.
- *Predefined Database Provisioning (PDBP)*: It is an automatic device provisioning setup where a script is constructing requests for devices based on a predefined database which contains all the devices information. Thus, the script is going through the database and provisions one by one devices to the FIWARE IoT Agent component. This solution is also automatic and working on the fly, however it needs a lot of computing power and a priori knowledge of the IoT devices connecting to the network. Moreover there is inconsistency between the synchronization of provisioning and devices joining the network due to the reason that the devices are provisioned without the confirmation that they have joined the network.

B. Performance Metrics

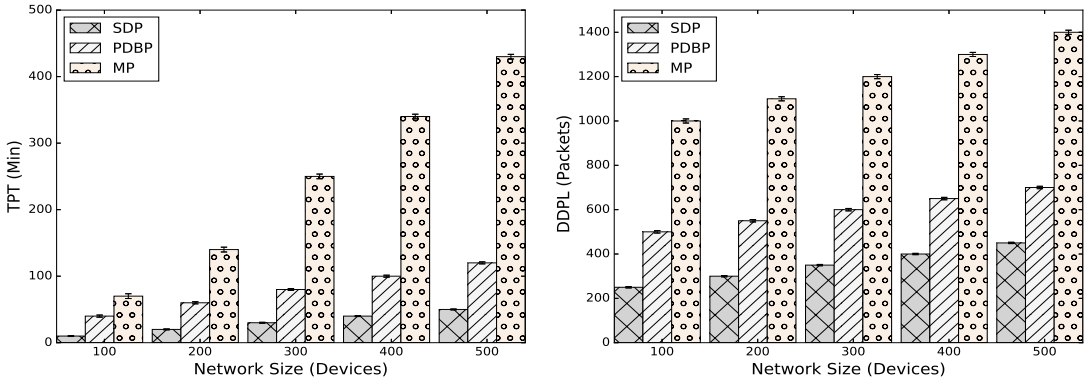
To evaluate the performance of the three described provisioning methods, the following metrics are defined:

- **[Total Provisioning Time] (TPT)** for all the devices: The definition of TPT describes the time required for all the network devices to be provisioned to the FIWARE back-end. This metric facilitates a general view of the deployment evaluation since it illustrates the whole provisioning efficiency.
- **[Time Inserted - Time Provisioned] (TITP)**: The difference between the device insertion time to the network and the provision time of the device. Measuring the time elapsed between these two time-stamps generates a metric TITP that describes the UX. The time required to provision the devices and enable them to send data reflects the user waiting time to retrieve data. By reducing the TITP time we are able to deliver data faster to the end user application.



(a) Latency Performance Evaluation (b) User Experience Performance Evaluation

Fig. 3: User Experience and Latency for SDN IoT integration performance



(a) Total Provisioning Time Evaluation (b) Reliability Performance Evaluation

Fig. 4: Total Provisioning Time and Reliability Performance

- **[Time Inserted - Data Received] (TIDR)**: Another metric called TIDR is used to evaluate the latency of the data transmission of the devices. The correlation of the insertion time of each device with the time of the first one hundred packets received from the FIWARE platform allows the analysis of latency between the device and the end-application.
- **[Deployment Duration - Packet Loss] (DDPL)**: DDPL is the metric evaluating network reliability. As Deployment Duration we define the time required for all the devices to be connected and ready to send data to the IoT platform. From the time the devices join the network until the end of the experiment they transmit 5000 User Datagram Protocol (UDP) packets. Monitoring the packet loss during this experiment time presents a clear picture about the reliability performance.

VI. Performance Evaluation

In order to evaluate the benefits of the proposed SDP framework, a series of experiments are conducted in different IoT network distributions aiming better performance of scalability, latency and UX. This section presents performance evaluation between three considered device provisioning methods such as Manual Provisioning (MP), Predefined Database Provisioning (PDBP) and the proposed SDP provisioning (SDP).

Based on aforementioned setup, the experiments emulate IoT deployments where the number of experiments are one hundred and for each case the maximum devices are five hundred. To derive the confidence interval, the replication method with 95% confidence interval was used. The device initialization happens continuously and the application is operating in real time. The experiments present the improvement on all the metrics used for evaluation thus show the advantages of using SDN within an IoT network.

A. Latency Evaluation

In Figures 3a and 3b multiple experiments run comparing MP, PDBP with SDP. The joining time of the devices is the same in all the experiments and the IoT data protocol is MQTT for all the experiments. Figure 3a presents the average latency results throughout multiple experiments and specifically for 500 emulated devices. For the valid Latency

evaluation we define a metric named Device Join Time which described the precise time that one IoT devices connects to a network which will send data. The correlation between the device join time within the network and the first one hundred packets received from FIWARE is the metric evaluating the latency performance. As shown in all our experiments SDP outperforms MP and PDBP by decreasing the latency.

B. User Experience Evaluation

Figure 3b demonstrates the UX analysis, the experiments are based on the same setup as described before and the units are also on average. The comparison between the join time and the provision time assesses the analysis of the UX performance. The time that the user needs to wait until the device is able to send data is critical since it affects the end user application performance. Also the end-to-end communication is a main indicator of the UX for facilitating end user applications collecting and analyzing the information. The SDP solution is again capable to reduce the provision time in seconds, by providing a plug and play feature to the deployment. The reason that the database provisioning is slower due to the processing time it needs to read and write the devices information.

C. Total Provisioning Time Evaluation

In Figure 4a the TPT is depicted in order to visualize the whole experimentation performance analysis. The samples in this evaluation are deployments of five hundred devices, where in all MP, PDBP and SDP provisioning mechanism tested the joining frequency is the same. As highlighted in figure 4a the SDN provisioning application performs much faster than the database solution in all samples tested, more precisely it reduces the total time almost to one third.

D. Reliability Evaluation

The packet loss is critical in IoT applications especially in real time data. To evaluate the reliability we compare the packet loss of the first 5000 packets transmitted with different network sizes during the deployment duration (DDPL). As shown in Figure 4b the SDP framework is performing better than the other configurations up to 40%. This is due to the reason that once the devices are connected to the network they will automatically start transmitting packets, therefore the provisioning time is critical to packet loss. Since SDP is reducing the total provisioning time required for all the network devices the initial packets will be delivered faster to the IoT platform.

VII. CONCLUSION

We present SDP a novel IoT device provisioning architecture based on software defined networks. Implemented as a framework that provides scalability to large WSN deployments. SDP is enhancing the current provisioning system leveraging SDN, and empowering IoT platforms with additional networking information. This approach enables a more dynamic management and monitoring of the IoT devices. Improved latency, reliability, and UX is the target of our implementation and performance evaluation. This framework highlights the advantages of IoT and SDN integration, focusing on providing scalability to future IoT deployments. We plan as future work to expand the research on SDN and IoT in the direction of developing a full stack framework tested with state of the art hardware. To further test the proposed mechanism we plan to compare it with other provisioning methods (i.e provisioning the devices within the FIWARE IoT platform).

ACKNOWLEDGMENT

This paper was performed under the REPLICATE project funded by the HORIZON 2020 the EU Framework Programme for Research and Innovation.

REFERENCES

[1] Internet of Things in 2020: A Roadmap for Future, RFID Working Group of The European Technology Platform on Smart System Integration (EPOSS), September 2008.
[2] 5G PPP, “View on 5G Architecture,” White Paper, July 2016. https://5g-ppp.eu/white-papers/
[3] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network, Study on new radio access technology; radio access architecture and interfaces, 3GPP TR 38.801 V2.0.0 (2017-03).
[4] IoT Market, Markets and Markets, http://www.marketsandmarkets.com/Market-Reports/iot-iam-market-67542546.html, Accessed in September 2017.
[5] M. R. Palattella et al., “Internet of Things in the 5G era: Enablers architecture and business models”, IEEE Journal on Selected Areas in Communications, vol. 34, no. 3, pp. 510-527, Mar. 2016.
[6] W. Ejaz et al., “Internet of Things (IoT) in 5G Wireless Communications”, in IEEE Access, vol. 4, no. , pp. 10310-10314, 2016.
[7] X. Lin, J.G. Andrews, A. Ghosh, R. Ratasuk, “An overview on 3GPP device-to-device proximity services,” IEEE Commun. Mag. 52 (4), 4048 (2014)
[8] L. Militano, G. Araniti, M. Condoluc, I. Farris, A. Iera, “Device-to-Device Communications for 5G Internet of Things,” EAI Endorsed Transactions on Internet of Things 15(1): e4, 2015.
[9] Zoltan Kis, Provisioning IoT with Web NFC, http://events.linuxfoundation.org/sites/events/files/slides/ProvisioningIoTwithWebNFC.pdf. Accessed in September 2017.
[10] Open Connectivity Foundation (OCF), https://openconnectivity.org/. Accessed in September 2017.
[11] Z. Qin, G. Denker, C. Giannelli, P. Bellavista and N. Venkatasubramanian, “A Software Defined Networking architecture for the Internet-of-Things”,2014 IEEE Network Operations and Management Symposium (NOMS), Krakow, 2014, pp. 1-9.
[12] Yaser JararwehEmail, Mahmoud Al-Ayyoub, Ala Darabseh, Elhadj Benkhelifa, Mladen Vouk, Andy Rindos,“SDIoT: a software defined based internet of things framework,” Journal of Ambient Intelligence and Humanized Computing,Volume 6, Issue 4, pp 453461, August 2015.
[13] Digicerts, Internet of Things: PKI Security for Smart Systems and Networked Devices. https://www.digicert.com/internet-of-things/provisioning-deployment.htm. Accessed in September 2017.
[14] Draft Standard for Local and Metropolitan Networks: Station and Media Access Control Connectivity Discovery, IEEE P802.1AB/D13, December, 2004.
[15] LLDP Media Endpoint Discovery Draft v2004-02, TR41.4-04-11-009, TIA ENGINEERING SUBCOMMITTEE, TR 41 CONTRIBUTION, 29 October, 2004.
[16] OpenDayLight, https://www.opendaylight.org/, Accessed in September 2017.
[17] FIWARE, https://www.fiware.org/, Accessed in September 2017.